Rethinking the KL Divergence in RLHF, From Single Sample to Mini-batch to Expectation

YiMing Liu[1]

[1]Email: letusgo126@126.com, From SYSU HCP Lab

March 2025

1 核心观点

用采样样本估计的梯度近似优化 KL 分布的梯度，二者近似等价需要满足条件为:

在相同的后验概率 $p_\theta(y|x)$ 的情况下 (θ 不动不更新，或者更新幅度极小，可以认为是 on-policy 采样)，采样到足够大的样本量。也就是说，有效样本量必须足够大。

	PPO	GRPO_on - policy	GRPO_off-policy	DPO	Diffusion_RLHF	Diffusion_DPO
θ 相同或偏差极小 (on - policy)	✓	✓	✓	×	✓	×
采样样本足够多	✓	?	✓	✓	×	×

2 贡献

- 强调 KL 估计和 KL 梯度估计的差异，只有梯度估计才更适合作为 Loss。在 appendix A提出了分类损失中 KL 散度梯度的无偏估计方法，section 3.1.2揭示了 k1 估计在 reward 函数中与 k2 估计作为 loss 之间的等价转换关系，并在 section 3.2从理论上证明了 k2 loss 相比 k3 loss 作为 KL 损失函数的优越性，k2 loss 梯度估计方法相较于 k3 loss 梯度估计方法具有更高的无偏性。在 section 3.2.4提供了 GRPO 中的 k3 loss 转换为 reward 函数中重参数化的形式。最后，从单纯 KL 估计角度，给出了 k3 估计的方差未必比 k1、k2 更小

的例子，反驳了持久以来认为 k3 估计存在绝对优越性的观点 [7, 8]。

KL 估计类型/是否适合作为	reward func	直接作为 loss
k1	✓	×
k2	×	✓
k3	×	有偏

- 在 section 3.1.4, 推导了 PPO 算法的原始奖励公式与优化目标，为策略优化提供了清晰的理论框架，澄清了现有文献中若干模糊表述。
- deepseek-math 中的 GRPO 仅在纯 on-policy 时才成立，结合重要性采样后的 off-policy 是不正确的，本文提供理论修正 section 3.3并给出具体代码实现https://github.com/OpenRLHF/OpenRLHF/pull/840。PS:GRPO_off-policy 和 PPO 一样，其实是一个近似的 on-policy 算法，rollout 大 batch 后用小 batch 训练并更新模型。
- 在 section 4.2中，提出了一种新颖的 KL 惩罚项集成方法，成功应用于 Diffusion RLHF 框架下的 DDPO 算法，解决了传统方法依赖外部数据集引入扩散预训练损失的局限性，显著降低了分布偏移风险。与 LLM 不同的是，LLM 基于词表直接计算 KL 散度相比采样是昂贵的，而 Diffusion 中直接计算 KL 散度的期望相比采样更加直接、简单省显存；且基于采样估计 Diffusion 的 KL 散度的梯度，做到有效的大 batch 几乎是不可能的。
- 在 section 4.3首次指出了当前 DiffusionDPO 存在的关键错误。

3 LLM RLHF

3.1 基于策略梯度方法的强化学习建模

为了简化表示，全部方法基于单步表示，改写成多步也完全成立。

3.1.1 RLHF 目标函数建模

基于人类反馈的强化学习（RLHF）的核心目标可建模为以下优化问题：

$$\arg\max_{\theta} J = \underbrace{\mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(y|x)}\left[r(x,y)\right]}_{\text{奖励最大化项}} - \beta\underbrace{\left[\mathbb{D}_{\mathrm{KL}}\left(\pi_\theta(y|x)\,||\,\pi_{\mathrm{ref}}(y|x)\right)\right]}_{\text{策略约束项}} \quad (1)$$

其中 x 为输入 prompt，y 为输出 answer。π_θ 为 LLM 在参数 θ 时的输出概率，是变量，可以进行参数优化。π_{ref} 为参考模型，不是变量，参数不优化。β 为温度系数，控制策略偏离参考模型 π_{ref} 的程度。该目标函数包含两个关键部分：奖励期望的最大化和 KL 散度约束。

3.1.2 目标函数分解与再合并

结合 KL 散度约束项 $\mathcal{L}_{\mathrm{KL}}(\pi_\theta, \pi_{ref})$，完整损失函数可表示为：

$$L_{total} = -\left(\mathcal{L}_R(\theta) - \beta \mathcal{L}_{\mathrm{KL}}(\pi_\theta, \pi_{ref})\right) \tag{2}$$

3.1.2.1 奖励最大化的损失函数 奖励项的损失函数为：

$$-\mathcal{L}_R(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(y|x)}\left[r(x,y)\right] = \mathbb{E}_{x\sim\mathcal{D}}\sum_y \left[\pi_\theta(y|x) r(x,y)\right] \tag{3}$$

求梯度后进行等价变换：

$$\begin{aligned}
-\nabla_\theta \mathcal{L}_R(\theta) &= \nabla_\theta \mathbb{E}_{x\sim\mathcal{D}} \sum_y \left[\pi_\theta(y|x) \cdot r(x,y)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}} \sum_y \left[\nabla_\theta \pi_\theta(y|x) \cdot r(x,y)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}} \sum_y \left[\pi_\theta(y|x) \frac{\nabla_\theta \pi_\theta(y|x)}{\pi_\theta(y|x)} \cdot r(x,y)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}} \sum_y \left[\pi_\theta(y|x) \frac{\nabla_\theta \pi_\theta(y|x)}{\pi_\theta(y|x)} \cdot r(x,y)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}} \sum_y \left[\pi_\theta(y|x) \nabla_\theta \log \pi_\theta(y|x) \cdot r(x,y)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}} \sum_y \left[\pi_\theta(y|x) \nabla_\theta \log \pi_\theta(y|x) \cdot r(x,y)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} \left[\nabla_\theta \log \pi_\theta(y|x) \cdot r(x,y)\right]
\end{aligned} \tag{4}$$

根据梯度，可以将奖励项的损失函数重写为更加常见的形式：

$$-\mathcal{L}_R(\theta) = \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\mathrm{old}}}(y|x)} \left[r(x,y) \log \pi_\theta(y|x)\right] \tag{5}$$

其中 θ_{old} 为采样时的模型参数，不是变量；θ 为 RL 训练时的模型参数，π_θ 是变量。纯 on-policy 时，θ_{old} 和 θ 二者在数值上相等。

实际计算中，由于 reward model 的限制，不可能对全词表的 y 计算 $r(x,y)$。仅能通过蒙特卡洛采样算法计算 $r(x,y_i)$ 后求取期望：

$$-\mathcal{L}_R(\theta) = \frac{1}{M \cdot N} \sum_{i=1}^{M} \sum_{j=1}^{N} \pi_{\theta_{\text{old}}}(y_j|x_i) \left[r(x_i, y_j) \log \pi_\theta(y_j|x_i) \right] \tag{6}$$

蒙特卡洛算法有效的要求是在模型参数几乎不变的情况下，样本量足够大，才能以样本估计和调整整个分布。

3.1.2.2 KL 惩罚的损失函数 KL 惩罚定义可以写为：

$$\text{KL}(\pi_\theta(y|x) \| \pi_{ref}(y|x)) = \mathbb{E}_{x\sim D, y\sim\pi_\theta(y|x)} \left[\log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} \right] \tag{7}$$

KL 惩罚梯度的推导具体推导见 eq. (81)，结果为：

$$-\nabla_\theta \text{KL}(\pi_\theta(y|x) \| \pi_{ref}(y|x)) = -\mathbb{E}_{x\sim D} \mathbb{E}_{y\sim\pi_\theta(y|x)} \left[\log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} \cdot \nabla_\theta \log \pi_\theta(y|x) \right] \tag{8}$$

在 reward function 中的 k1 估计，的损失函数形式 因此损失函数可推导为：

$$\mathcal{L}_{\text{KL}}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{x\sim D} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \cdot \log \pi_\theta(y|x) \right] \tag{9}$$

其中 y 为全词表。这个损失函数既可以单独使用，也可以如下文 eq. (25) 重参数化到 reward 中，但是要注意，如果不是全词表，我们的目标是优化分布，实际实践上是用蒙特卡洛采样估计并优化分布，这要求我们在模型参数基本不变的情况下采样到足够大的 batch_size。

k2 loss 的损失函数形式 KL 梯度

$$-\nabla_\theta \text{KL}(\pi_\theta(y|x) \| \pi_{ref}(y|x)) \tag{10}$$

$$= -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} \left[\log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \cdot \nabla_\theta \log \pi_\theta(y|x) \right] \tag{11}$$

$$
\begin{aligned}
&- \nabla_\theta \mathrm{KL}(\pi_\theta(y|x) \| \pi_{ref}(y|x)) \Big|_{\theta=\theta_{\text{old}}} && (12)\\
&= -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\nabla_\theta \log \pi_{\theta_{\text{old}}}(y|x) \cdot \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \right] && (13)\\
&= -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\nabla_\theta \log \pi_\theta(y|x)|_{\theta=\theta_{\text{old}}} \cdot \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \right] && (14)\\
&= -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\nabla_\theta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \Big|_{\theta=\theta_{\text{old}}} \cdot \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \right] && (15)\\
&= -\frac{1}{2} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\nabla_\theta \left(\log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \right)^2 \Big|_{\theta=\theta_{\text{old}}} \right] && (16)
\end{aligned}
$$

定义 k_2 loss 下的损失函数

$$
\mathcal{L}_{k_2}(\theta) = \frac{1}{2} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\text{old}}(y|x)} \left[\left(\log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \right)^2 \right] \tag{17}
$$

进一步验证，求 $\mathcal{L}_{k_2}(\theta)$ 损失函数的梯度:

$$
\begin{aligned}
-\nabla_\theta \mathcal{L}_{k_2}(\theta) &= -\frac{1}{2} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\text{old}}(y|x)} \left[\nabla_\theta \left(\log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \right)^2 \right] && (18)\\
&= -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\text{old}}(y|x)} \left[\log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \cdot \nabla_\theta \log \pi_\theta(y|x) \right] && (19)
\end{aligned}
$$

得出结论

$$
\nabla_\theta \mathcal{L}_{k_2}(\theta) = \nabla_\theta \mathrm{KL}(\pi_\theta(y|x) \| \pi_{ref}(y|x)) \tag{20}
$$

3.1.3 合并奖励和 KL 惩罚梯度

两个组成部分:

1. **奖励项梯度**:

$$
\nabla_\theta \mathcal{L}_R(\theta) = \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[r(x,y) \nabla_\theta \log \pi_\theta(y|x) \right] \tag{21}
$$

2. **KL 散度项梯度**（具体推导见引理 eq. (81)）:

$$
-\nabla_\theta \mathcal{L}_{\text{KL}} = -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{ref}(y|x)} \cdot \nabla_\theta \log \pi_\theta(y|x) \right] \tag{22}
$$

将二者结合可得总梯度:

$$
\begin{aligned}
-\nabla_\theta L_{total} &= \nabla_\theta \mathcal{L}_R - \beta \nabla_\theta \mathcal{L}_{\mathrm{KL}_t} \\
&= \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\left(r(x,y) - \beta \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{ref}(y|x)} \right) \nabla_\theta \log \pi_\theta(y|x) \right]
\end{aligned} \tag{23}
$$

3.1.4 等效奖励函数构造与 PPO 算法的 reward 函数

由公式 eq. (23) 可以重新定义新的优化目标:

$$
\arg\max_\theta J' = \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} \left[r(x,y) - \beta \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{ref}(y|x)} \right] \tag{24}
$$

该形式将 KL 正则项内化为奖励函数的组成部分，从而实现了优化目标的重新参数化。

提取出强化学习整个奖励部分，容易得出 PPO 算法相同的奖励函数为:

$$
\tilde{r}(y|x) = r(x,y) - \beta \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)} \tag{25}
$$

这个时候由于 $r(x,y)$ 的限制并且 KL 项合并到一起了，KL 计算也不能在全词表了，整个式子仅能使用蒙特卡洛估计，所以有效样本量一定要足够大。

3.2 KL 距离的数学性质分析

3.2.1 基于采样梯度估计分布梯度

KL 梯度的定义为:

$$
-\nabla_\theta \mathcal{L}_{\mathrm{KL}} = -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{ref}(y|x)} \cdot \nabla_\theta \log \pi_\theta(y|x) \right] \tag{26}
$$

其中会对 y 求期望，但在代码工程实践中，因为显存和计算量等限制，求取期望并不完全现实，往往会用蒙特卡洛采样等手段，采样 Mini-batch 后对模型参数进行随机梯度更新，最极端的情况下，我们甚至只能采样单样本点进行计算。

对此 KL 有不同的处理方式：1.LLM 本质上是一个分类器，它的类别终归是有限的 (1w∼3w)，遍历完整词表也可以做到，直接就能计算得到 KL

分布：

$$-\nabla_\theta \mathcal{L}_{\mathrm{KL}} = -\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{vocab\_size} \pi_{\theta_{\mathrm{old}}}(y_i|x) \left[\log \frac{\pi_{\theta_{\mathrm{old}}}(y_i|x)}{\pi_{ref}(y_i|x)} \cdot \nabla_\theta \log \pi_\theta(y_i|x)\right] \tag{27}$$

当然，也可以直接带入到定义中去计算，也能起到相同效果。这种方法的好处是即便 batch_size=1，只有一个样本，计算的也是完整的 KL 散度和 KL 散度的梯度。

2. 因为词表过大，在更多的工程实践中，人们往往会用蒙特卡洛采样的手段，通过在有限样本上近似估计 KL 散度及梯度，再进行 Mini-batch 更新：

$$-\nabla_\theta \mathcal{L}_{\mathrm{KL}} = -\frac{1}{N}\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{N} \pi_{\theta_{\mathrm{old}}}(y_i|x) \left[\log \frac{\pi_{\theta_{\mathrm{old}}}(y_i|x)}{\pi_{ref}(y_i|x)} \cdot \nabla_\theta \log \pi_\theta(y_i|x)\right] \tag{28}$$

其中 N 是采样次数。理论上，只要采样数量足够大，采样梯度也能够很好近似 KL 分布的梯度。这种方案每次只要记录下采样到的 y_i 的概率值，显存消耗少。但问题是它有两个要求，一个是采样样本量足够大，另一个是采样样本时模型参数不能变化。这其实是很难满足的，采样样本量足够大且参数不能变化，这意味着采样次数多，更新次数少，训练效率不高。所以在工程实践中，我们往往都是 Mini-batch，它是单样本和期望上的一个中间状态。也就是说，在分析时，我们要同时考虑单样本和期望两种情况。（单样本指的是 bs=1，采样后就更新模型参数；期望指的是采样无穷样本后在更新参数）

下面我将介绍基于采样梯度估计 KL 分布梯度的具体做法和性质，KL 估计主要定义为 k1、k2、k3[7]：

3.2.2 何种 KL 估计适合直接作为损失函数

k1 估计 k1 直接作为 loss 对应的损失函数形式为：

$$\begin{aligned}\mathcal{L}_{\mathrm{KL\_k1\ loss}}(\pi_\theta, \pi_{ref}) &= \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_{\theta_{\mathrm{old}}}(y|x)}(\log \pi_\theta(y|x) - \log \pi_{ref}(y|x)) \\ &= \mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{batch\_size} \pi_\theta(y_i|x)(\log \pi_\theta(y_i|x) - \log \pi_{ref}(y_i|x))\end{aligned} \tag{29}$$

其梯度表达式为：

$$-\nabla_\theta \mathcal{L}_{\text{KL\_k1 loss}} = -\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{batch\_size} \pi_\theta(y_i|x)\nabla_\theta \log \pi_\theta(y|x) \tag{30}$$

- **单样本**：

$$-\nabla_\theta \mathcal{L}_{\text{KL\_k1 loss}} = -\mathbb{E}_{x\sim\mathcal{D}}(\pi_\theta(y_i|x)\nabla_\theta \log \pi_\theta(y_i|x)) \tag{31}$$

  该梯度恒指向降低当前策略概率的方向，采样到样本就降低生成概率。会导致策略网络的概率输出持续衰减，最终引发模型崩溃问题。

- **期望**：对梯度求期望：

$$\begin{aligned}
-\nabla_\theta \mathcal{L}_{\text{KL_k1 loss}} &= -\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{\infty} \pi_\theta(y_i|x)\nabla_\theta \log \pi_\theta(y_i|x) \\
&= -\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{\infty} \pi_\theta(y_i|x) \frac{\nabla_\theta \pi_\theta(y_i|x)}{\pi_\theta(y_i|x)} \\
&= -\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{\infty} \nabla_\theta \pi_\theta(y_i|x) \\
&= -\mathbb{E}_{x\sim\mathcal{D}} \nabla_\theta \sum_{i=1}^{\infty} \pi_\theta(y_i|x) \\
&= -\mathbb{E}_{x\sim\mathcal{D}} \nabla_\theta 1 = 0
\end{aligned} \tag{32}$$

  在期望上模型梯度为 0，就是说样本数量特别大的时候，k1 作为 loss 对模型更新几乎无影响。

- 综上，无论哪种情况，都不是在估计 KL 散度分布的梯度，并且在 batch 小时，大概率会崩溃。

k2 估计

$$\begin{aligned}
-\nabla_\theta \mathcal{L}_{\text{KL_k2 loss}} &= -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} \left[\log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{ref}(y|x)} \cdot \nabla_\theta \log \pi_\theta(y|x) \right] \\
&= -\mathbb{E}_{x\sim\mathcal{D}} \sum_{i=1}^{batch_size} \pi_{\theta_{\text{old}}}(y_i|x) \left[\log \frac{\pi_{\theta_{\text{old}}}(y_i|x)}{\pi_{ref}(y_i|x)} \cdot \nabla_\theta \log \pi_\theta(y_i|x) \right]
\end{aligned} \tag{33}$$

如 eq. (17) 所示，我们以及推导出了 k2 loss 并且它在期望上是有效的，通过大 btach_size 采样，是非常有效的，能够很好起到限制策略模型和参考

模型的 KL 距离的作用，防止策略模型崩溃掉。但必须强调的是，在单样本或低 btach_size 下，使用大学习率也容易崩溃，但单样本梯度如下所示：

$$
\begin{aligned}
-\nabla_\theta \mathcal{L}_{\text{KL\_k2 loss}} &= -\mathbb{E}_{x\sim\mathcal{D}}\pi_{\theta_{\text{old}}}(y_i|x)\left[\log\frac{\pi_{\theta_{\text{old}}}(y_i|x)}{\pi_{ref}(y_i|x)}\cdot\nabla_\theta\log\pi_\theta(y_i|x)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\pi_{\theta_{\text{old}}}(y_i|x)\left[(\log\pi_{ref}(y_i|x)-\log\pi_{\theta_{\text{old}}}(y_i|x))\cdot\nabla_\theta\log\pi_\theta(y_i|x)\right]
\end{aligned}
\tag{34}
$$

当 $\log\pi_{ref}(y_i|x) > \log\pi_{\theta_{\text{old}}}(y_i|x)$,$-\nabla_\theta\mathcal{L}_{\text{KL}} < 0$, 会降低 $\log\pi_{\theta(y_i|x)}$ 生成概率。反之亦然。也就是说单样本尽管没有起到限制 KL 的作用，但是它对这个样本的类别 y_i 起到了一定限制作用，防止它偏离参考模型过远。所以在小学习率和小样本下，再配合 eq. (25) 中的 β 小系数，似乎工作得也不错。但我们仍要注意，在一些纯 on-policy 算法上，比如 GRPO，deepseek 公司有 2k 张 A100/H800 这种显卡，能够把 batch_size 开到 1k∼2k, 再配合梯度累积技术，可能不会碰到超小 batch 采样得问题，能够很好近似 KL 分布的梯度，非常有效。但个人研究者手中可能没有那么多显卡，batch 量可能非常非常小，这个时候建议结合梯度累积技术，或者使用一些结合重要性采样的 on-policy 算法（RF++/GRPO_off-policy,GRPO_off-policy 后文会提到），这些算法 rollout 的 batch 量会相对可观，能够有更大的有效样本量，能够更好起到真正 KL 惩罚的作用。

k3 估计量的近似特性 k3 构造的损失函数具有特殊形式：

$$
\mathcal{L}_{\text{KL\_k3 loss}} = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)}\left(\frac{\pi_{ref}(y|x)}{\pi_\theta(y|x)}-\log\frac{\pi_{ref}(y|x)}{\pi_\theta(y|x)}-1\right) \tag{35}
$$

对应的梯度表达式揭示其近似本质：

$$
-\nabla_\theta\mathcal{L}_{\text{KL\_k3 loss}} = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)}\left[\left(\frac{\pi_{\text{ref}}(y|x)}{\pi_{\theta_{\text{old}}}(y|x)}-1\right)\nabla_\theta\log\pi_\theta(y|x)\right] \tag{36}
$$

令 $x=\frac{\pi_{\text{ref}}(y|x)}{\pi_{\theta_{\text{old}}}(y|x)}$，k2 loss 与 k3 loss 的梯度可对比表示为：

$$
\text{k2 loss 梯度}:\log x\cdot\nabla_\theta\log\pi_\theta \tag{37}
$$

$$
\text{k3 loss 梯度}:(x-1)\cdot\nabla_\theta\log\pi_\theta \tag{38}
$$

在策略邻域 ($x\approx 1$) 进行泰勒展开时，$\log x\approx x-1$，此时 k3 梯度构成 k2 梯度的线性近似。但这种近似具有两个关键缺陷：1. **有偏性**：当策略显著偏离参考策略 (x 远离 1, 训练后期 π_{ref} 离 $\pi_{\theta_{\text{old}}}$ 较远，尤其是

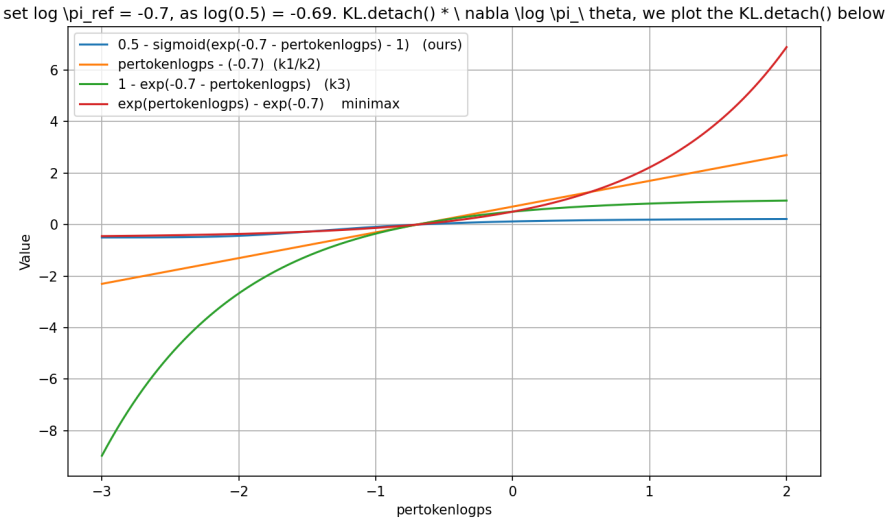

图 1: k1/k2 和 k3 及 MiniMax-01[5] 实现的 KL 梯度系数项对比，代码见 appendix G，相比 k1/k2,k3 在模型相比参考模型概率低很多时，数值会发生指数级变化，造成不稳定。(图中设定 $\log \pi_{ref} = -0.7$，此时概率 $\pi_{ref} \approx 0.5$)

$\pi_{ref} >> \pi_{\theta_{\text{old}}}$) 时 (模型输出的类别为低概率范围)，近似误差呈非线性增长，可视化见 fig. 3。2. **非对称性**：$x - 1$ 对 $\pi_{\theta_{\text{old}}} > \pi_{ref}$ 和 $\pi_{\theta_{\text{old}}} < \pi_{ref}$ 的响应特性不对称。所以 k3 估计的 loss 仅仅是 k2 loss 的近似，可以明确证明，k3 loss 比 k2 loss 更好！实验中也容易观察到使用 k3 loss 的 GRPO 算法比用 k2 loss 的方差波动更大。

3.2.3 何种 KL 估计适合放在奖励函数中

在 PPO 算法设计中，仅特定形式的 KL 散度估计量适合作为奖励函数的惩罚项。根据理论分析，k1 估计量具有数学适用性，其表达式如 eq. (25) 所示。在 eq. (9) 我们已经推导出它了，并且它与 k2 loss 是等价的 eq. (20)。它的性质在 k2 loss 中讨论。

然而，k2 和 k3 估计量无论如何均不适合作为惩罚项，原因如下：考虑 k2 和 k3 的估计值 kl 具有恒正特性，其梯度方向始终满足：

$$-\nabla_\theta \text{KL}(\pi_\theta \| \pi_{ref}) = -\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_{\theta_{\text{old}}}(y|x)} kl \nabla_\theta \log \pi_\theta(y|x) \tag{39}$$

此时无论 $\pi_\theta(y|x)$ 与 $\pi_{ref}(y|x)$ 的概率分布关系如何，$-\nabla_\theta \text{KL}(\pi_\theta \| \pi_{ref})$

图 2: 不同 KL 的实验，GRPO 算法，batch=12，group_size=6, 基于 x_r1 框架 [2]，$\beta = 0.04$

衡小于 0，梯度更新方向都会强制降低当前策略 π_θ 生成任意动作 y 的概率。这种单向的惩罚机制会导致模型崩溃，无论是单样本采样还是期望。

3.2.4 Reward 函数中的 KL 估计和 KL loss 是可以相互转换的

KL 估计主要定义为 k1、k2、k3[7]，正如 eq. (25) 的 reward 中的 k1 估计部分 $-\beta \log \frac{\pi_{\theta_{\text{old}}}(y|x)}{\pi_{\text{ref}}(y|x)}$ 来自于 eq. (9)，既可以单独放在外面，也可以重参数化到 reward 中，同时如 eq. (17) 还完全等价于 k2 估计作为 loss。

结合 eq. (21) 和 eq. (36), 可以得出 k3 loss 重参数化到 reward 函数中，易得:

$$\tilde{r}(y|x) = r(x,y) - \beta \left(1 - \frac{\pi_{\text{ref}}(y|x)}{\pi_{\theta_{\text{old}}}(y|x)}\right) \tag{40}$$

3.3 GRPO_off-policy

GRPO_on-policy 在个人显卡比较少的时候，即便开启梯度累积，有效 batch_size 都比较小。解决的办法是引入 PPO-2 的重要性采样和 clip，变成 PPO 一样的 off-policy 算法，就可以增大 rollout_batch，rollout_batch 是在模型参数不变的情况下分批采样出来的，有效 batch_size 可以做到非常大。

GRPO 原论文中给出的公式为：

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$
$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}, 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{i,t}\right] - \beta \mathrm{D}_{KL}[\pi \tag{41}$$

参考 [13] 简化表示为：

$$\mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}A - KL(\pi_\theta, \pi_{\text{ref}})\right] = \mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}A - \mathbb{E}_{\pi_\theta}(\log\pi_\theta - \log\pi_{\text{ref}})\right] \tag{42}$$

因此，当将 KL 项用作损失时，需要保留“完整词汇表（vocab_size）”并在整个词表上计算。

如果依旧只想保留一个类别并使用蒙特卡洛方法用采样梯度近似去调整整个分布时，TRL[11]、OpenRLHF[4] 和 Verl[9] 代码中的“compute_approx_kl”函数实现：

$$\mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}A - \mathbb{E}_{\pi_{\theta_{\text{old}}}}(\log\pi_\theta - \log\pi_{\text{ref}})\right] \tag{43}$$

由于我们实际上使用 $\pi_{\theta_{\text{old}}}$ 进行采样，并用 π_θ 去更新，除非是完全 on-policy 的状况，否则会导致丢失 π_θ 的导数信息，原始实现仅在 on-policy 时成立，像 PPO 那种 off-policy 的话 GRPO 的原论文写法是有问题的。

除了使用全词表、完全 on-policy 外，在实现 off-policy 时，GRPO 必须修正，而修正的方法很简单，就是把 KL 散度的梯度估计也加上重要性采样和 PPO2-clip，合并同类项后，等同于把 KL 散度像 PPO 一样藏到 Reward 里面去：

等效 k1 reward/k2 loss 估计（更优）：$\hat{R}_{i,t} = (\hat{A}_{i,t} - \beta\log\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{ref}(o_{i,t}|q,o_{i,<t})})$

等效 k3 loss 估计（有偏）：$\hat{R}_{i,t} = \left(\hat{A}_{i,t} - \beta\left(1 - \frac{\pi_{\text{ref}}(y|x)}{\pi_{\theta_{\text{old}}}(y|x)}\right)\right)$

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$
$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}\hat{R}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}, 1-\varepsilon, 1+\varepsilon\right)\hat{R}_{i,t}\right]\right\}. \tag{44}$$

https://github.com/OpenRLHF/OpenRLHF/pull/840/files已经提交给OpenRLHF。从这个角度，GRPO 独特性可能仅存在于 Group Norm 了。

总结起来，1) 巨大算力和显存在完整词表计算，2) 完全 on-policy 堆巨量硬件大 batch，3) 使用 GRPO_off-policy。

3.4 Group Norm 的改进

Group Norm 是 GRPO 独有的: 同一个 prompt 生成 G 个 answer，用 reward model 打分后得到 group 的奖励序列 $\mathbf{r} = \{r_1, r_2, \cdots, r_G\}$, 然后对 group 的 reward 进行如下归一化得到优势 $\hat{A}_{i,t}$:

$$\hat{A}_{i,t} = \widetilde{r}_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}, \tag{45}$$

但 GRPO 在非 rule-based reward 上往往容易崩溃，这大概率是因为其 Group Norm 中/std() 的操作导致的: 比如 reward_list = [0.99999,1.00001,0.99999,1.00001], Adv_list = [-0.8660, 0.8660, -0.8660, 0.8660]。也就是说，方差非常小，完全可能是数值误差，会被 Group Norm 放大到一个很夸张的程度。

想要避免出现方差过小这种极端情况，建议进行 clip 操作，为此我们引入 clip_std 函数: $clip\_std() = max(min(std(), max), min)$, 此时优势计算函数变为:

$$\hat{A}_{i,t} = \widetilde{r}_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{clip\_std}(\mathbf{r})}, \tag{46}$$

PS: 当 reward model 的打分在 0.0-1.0 间时，Group Norm 的 reward_list 的 std 方差一定是 <1 的 appendix F，起到的作用一定是增大区分度。并且 reward_list 方差越小，放大程度越大，区分度增加越明显。

3.5 DPO 的问题

DPO 理论上应该优化的是 eq. (103), 它是一个分布；但在具体实现上，是通过采样样本估计并实现的 eq. (102)。用样本近似估计并调整分布，需

要满足我们的核心观点，有效样本量足够大。尽管 DPO 的 batch_size 足够大，但它优化分布的有效 batch_size 有可能会很小的，未必能很好满足蒙特卡洛估计的要求，这是因为 DPO 是 off-policy 的算法，难以控制训练数据的分布：

1）有很多工作可能会用外来数据（GPT4 收集）或者其它模型数据来训练 DPO 模型，其它模型的参数和当前模型参数可能差异巨大，训练数据和当前模型采样的数据分布可能差异巨大，直接代入 eq. (102)，良好估计并优化当年模型的分布 eq. (103) 可能非常困难。

2）即便是相同模型采样的样本训练当前模型，比如从最开始采样大量数据标注好，然后再训练 DPO 模型；DPO 模型参数会更新多次，更新后的分布和最开始采样数据时的分布不一致。也很难估计好并优化好当前模型的分布。

所以就会有 [16] 观察：相同模型采样的数据训练相同模型 chosen reward 至少还会上升，其它模型采样的数据训练当前模型 chosen reward 会下降。

解决的办法直观上很简单，要么改成一些近似 online 的算法，比如采样一轮数据标注后训练一轮，循环迭代；要么加一些 sft 的 loss（极端一点理解就是用其它模型数据蒸馏当前模型，使得两个模型变得一致），使得模型的分布和训练数据的分布更接近，起到一定近似的效果。

但 DPO 的优势就在于它是一个 off-policy 的算法，如果让它变得符合蒙特卡洛条件并用样本满足期望上分布的调整，比如改成 online 算法，会使得 DPO 的优势可能不复存在，infra 上也可能会非常复杂，这也确实是 DPO 进一步改进的困难之处。

总之，DPO 最终的目标是优化分布，但实践上只能是 Mini-batch 样本。训练数据和模型之间必须符合或近似符合蒙特卡洛条件，才能起到最好的优化效果。

4 Diffusion RLHF

4.1 Diffusion RLHF 之 DDPO

下文 π 表示概率,Diffusion 其它论文常用符号为 p。

4.1.1 扩散模型基础

Denoising Diffusion Probabilistic Models (DDPM) 的逆向过程采样公式定义为：

$$\pi_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)=\mathcal{N}\left(\boldsymbol{x}_{t-1} ; \boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right), \sigma_t^2 \mathbf{I}\right) \tag{47}$$

其具体采样形式为：

$$\boldsymbol{x}_{t-1}=\boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)+\sigma_t \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{48}$$

其中 $\boldsymbol{c}$ 为条件信息，σ_t^2 为预定义的方差参数。

4.1.2 概率密度与梯度推导

当方差固定时，高维高斯分布的概率密度函数可展开为：

$$\pi_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)=\frac{1}{\left(2 \pi \sigma_t^2\right)^{d / 2}} \exp \left(-\frac{\left\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)\right\|^2}{2 \sigma_t^2}\right) \tag{49}$$

其中 d 为潜变量维度。取对数概率得：

$$\log \pi_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)=-\frac{d}{2} \ln (2 \pi)-\frac{d}{2} \ln \sigma_t^2-\frac{\left\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)\right\|^2}{2 \sigma_t^2} \tag{50}$$

对模型参数 θ 求梯度时，前两项为常数项，故梯度表达式简化为：

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)=-\frac{1}{2 \sigma_t^2} \nabla_{\boldsymbol{\theta}}\left\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)\right\|^2 \tag{51}$$

4.1.3 Diffusion 强化学习目标函数

DDPO[1] 的优化目标定义为:

$$\arg \max _\theta J=\underbrace{\mathbb{E}_{c \sim \mathcal{D}, x_0 \sim \pi_\theta(\cdot \mid \cdot, c)}\left[r(x_0, \boldsymbol{c})\right]}_{\text{奖励最大化项}} \tag{52}$$

DDPO 的梯度函数定义为：

$$\nabla_{\boldsymbol{\theta}} J=\nabla_{\boldsymbol{\theta}} \mathcal{L}_{DDPO}=\mathbb{E}_{\boldsymbol{c} \sim D, t \sim U(0, T)}\left[r\left(\boldsymbol{x}_0, \boldsymbol{c}\right) \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)\right] \tag{53}$$

展开后得到完整损失函数表达式：

$$\begin{aligned}
-\mathcal{L}_{DDPO} &= \mathbb{E}_{\boldsymbol{c} \sim D, t \sim U(0, T)}\left[r\left(\boldsymbol{x}_0, \boldsymbol{c}\right) \log \pi_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{c} \sim D, t \sim U(0, T)}\left[r\left(\boldsymbol{x}_0, \boldsymbol{c}\right)\left(\underbrace{-\frac{d}{2} \ln (2 \pi)-\frac{d}{2} \ln \sigma_t^2}_{\text{常数项}}-\frac{\left\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\right\|^2}{2 \sigma_t^2}\right)\right]
\end{aligned} \tag{54}$$

注意到常数项对优化无贡献，可得等价损失函数：

$$-\mathcal{L}_{DDPO_{equiv}} = \mathbb{E}_{\boldsymbol{c}\sim D, t\sim U(0,T)}\left[-r\left(\boldsymbol{x}_0, \boldsymbol{c}\right)\frac{\|\boldsymbol{x}_{t-1} - \boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)\|^2}{2\sigma_t^2}\right] \tag{55}$$

4.1.4 与预训练目标的关联

对比扩散模型预训练目标：

$$-\mathcal{L}_{pretrain}(\theta) = \mathbb{E}_{(\mathbf{x}_0,\mathbf{c})\sim p(\mathbf{x}_0,\mathbf{c}), t\sim\mathcal{U}\{0,T\}, \mathbf{x}_t\sim q(\mathbf{x}_t|\mathbf{x}_0)}\left[-\frac{\|\tilde{\boldsymbol{\mu}}(\mathbf{x}_0, t) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{c}, t)\|^2}{2\sigma_t^2}\right] \tag{56}$$

可知 DDPO 在预训练目标基础上引入了奖励加权机制，实现对齐优化。

4.1.5 重要性采样扩展

引入重要性采样比后，DDPO 改进形式为：

$$\begin{aligned}-\mathcal{L}_{DDPO_{IS}} &= \mathbb{E}_{\boldsymbol{c}\sim D, t\sim U(0,T)}\left[r\left(\boldsymbol{x}_0, \boldsymbol{c}\right)\frac{p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)}{p_{\boldsymbol{\theta}_{\text{old}}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c}\right)}\right] \\ &= \mathbb{E}_{\boldsymbol{c}\sim D, t\sim U(0,T)}\left[r\left(\boldsymbol{x}_0, \boldsymbol{c}\right)\exp\left(\frac{\|\mathbf{x_{t-1}} - \boldsymbol{\mu}_{\theta_{\text{old}}}\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)\|^2}{2\sigma_t^2} - \frac{\|\mathbf{x_{t-1}} - \boldsymbol{\mu}_\theta\left(\boldsymbol{x}_t, \boldsymbol{c}, t\right)\|^2}{2\sigma_t^2}\right)\right]\end{aligned} \tag{57}$$

其中 θ_{old} 为旧策略参数，通过重要性采样比 $\frac{p_\theta}{p_{\theta_{\text{old}}}}$ 实现策略的稳定更新。可进一步采用 PPO-2 clip 策略稳定训练。

实现细节说明 注：在理论推导中，L2 范数平方对应的损失函数采用总和归约（reduction='sum'），而实际代码实现中通常采用均值归约（reduction='mean'）以增强数值稳定性。在 DDPM 中，这种理论推导 [3] 与工程实践 [10] 的差异性的标准实现中同样存在，其主要差异体现在梯度幅值的缩放比例，而不会改变优化方向。具体表现为：

- 理论推导：$\|\cdot\|^2$ 对应元素级平方和（sum）
- 工程实现：$\frac{1}{d}\|\cdot\|^2$ 对应元素级平方均值（mean）

其中 d 为特征维度，这种缩放操作通过保持梯度量级在合理范围内，有效避免了训练过程中的数值溢出问题。以上推导中的 μ 替换成 DDPM 中的 ϵ、Rectified Flow 中的 v 和 score sde 中的 score 都成立，推导省略。

4.2 基于参考模型的扩散强化学习对齐方法 (DDPO with Reference Model)

$$\arg\max_{\theta} J = \underbrace{\mathbb{E}_{c\sim\mathcal{D},x_0\sim\pi_\theta(\cdot|c)}\left[r(x_0,c)\right]}_{\text{奖励最大化项}} - \beta\underbrace{\left[\mathbb{D}_{\mathrm{KL}}\left(\pi_\theta(x_{t-1}|x_t,c)\parallel\pi_{\mathrm{ref}}(x_{t-1}|x_t,c)\right)\right]}_{\text{策略正则化项}} \tag{58}$$

根据 appendix B的推导结果，KL 散度的期望可以显式表达为：

$$\mathcal{L}_{DDPO_{KL}} = \mathbb{E}_{c\sim\mathcal{D},t\sim U(0,T)}\left[\frac{\|\boldsymbol{\mu}_\theta(x_t,c,t)-\boldsymbol{\mu}_{\mathrm{ref}}(x_t,c,t)\|^2}{2\sigma_t^2}\right] \tag{59}$$

$$-\nabla\boldsymbol{\mu}_\theta(x_t,c,t)\mathcal{L}_{DDPO_{KL}} = \mathbb{E}_{c\sim\mathcal{D},t\sim U(0,T)}\left[\frac{\left(\boldsymbol{\mu}_{\mathrm{ref}}(x_t,c,t)-\boldsymbol{\mu}_\theta(x_t,c,t)\right)}{\sigma_t^2}\right] \tag{60}$$

与分类模型直接计算 KL 散度的期望需要在全词表上计算成本和显存消耗特别大不同，Diffusion 模型直接计算 KL 散度的期望是非常直接且比蒙特卡洛采样估计更加简单。

因此，完整的 DDPO 目标函数可分解为以下形式：

$$\begin{aligned}
\mathcal{L}_{DDPO_{total}} &= -\left(\mathcal{L}_{DDPO} - \beta\mathcal{L}_{DDPO_{KL}}\right) \\
&= -\mathbb{E}_{c\sim D,t\sim U(0,T)}\Big[r(x_0,c)\log\pi_\theta(x_{t-1}|x_t,c)\Big] \\
&\quad + \beta\mathbb{E}_{c\sim\mathcal{D},t\sim U(0,T)}\left[\frac{\|\boldsymbol{\mu}_\theta(x_t,c,t)-\boldsymbol{\mu}_{\mathrm{ref}}(x_t,c,t)\|^2}{2\sigma_t^2}\right] \\
&= \mathbb{E}_{c\sim D,t\sim U(0,T)}\left[\frac{r(x_0,c)}{2\sigma_t^2}\|x_{t-1}-\boldsymbol{\mu}_\theta(x_t,c,t)\|^2\right. \\
&\quad \left. + \frac{\beta}{2\sigma_t^2}\|\boldsymbol{\mu}_\theta(x_t,c)-\boldsymbol{\mu}_{\mathrm{ref}}(x_t,c,t)\|^2\right]
\end{aligned} \tag{61}$$

4.3 采样估计 Diffusion RLHF 中 KL 分布惩罚是困难且昂贵的

若直接带入 PPO 的奖励函数（如 eq. (25) 所示），将推导出完全基于采样的 Diffusion RLHF 的损失函数：

$$\begin{aligned}\mathcal{L}_{DDPO_{sampling}} &= -\mathbb{E}_{\boldsymbol{c}\sim D,t\sim U(0,T)}\Bigg[\left(r\left(\boldsymbol{x}_0,\boldsymbol{c}\right)-\beta\log\frac{\pi_{\boldsymbol{\theta}_{\text{old}}}\left(\boldsymbol{x}_{t-1}\mid\boldsymbol{x}_t,\boldsymbol{c}\right)}{\pi_{\boldsymbol{ref}}\left(\boldsymbol{x}_{t-1}\mid\boldsymbol{x}_t,\boldsymbol{c}\right)}\right)\\&\qquad\times\log\pi_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1}\mid\boldsymbol{x}_t,\boldsymbol{c}\right)\Bigg]\\&=\mathbb{E}_{\boldsymbol{c}\sim D,t\sim U(0,T)}\Bigg[\left(r\left(\boldsymbol{x}_0,\boldsymbol{c}\right)-\beta\left(\frac{\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{ref}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2}{2\sigma_t^2}-\frac{\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{\theta_{\text{old}}}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2}{2\sigma_t^2}\right)\right)\\&\qquad\cdot\frac{\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{\theta}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2}{2\sigma_t^2}\Bigg]\end{aligned}\tag{62}$$

$\mathcal{L}_{DDPO_{sampling}}$ 中的 KL 惩罚项，它是基于采样的 KL 散度梯度的估计：

$$\begin{aligned}&-\mathcal{L}_{KL_{sampling}}\\&=\frac{\beta}{4\sigma_t^4}\mathbb{E}_{\boldsymbol{c}\sim D,t\sim U(0,T)}\Big[\left(\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{ref}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2-\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{\theta_{\text{old}}}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2\right)\\&\qquad\cdot\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{\theta}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2\Big]\end{aligned}\tag{63}$$

它对应 LLM RLHF 中的 k1 在 reward 函数中的损失函数形式 eq. (9)，它也完全等价于 k2 loss 的损失函数形式 eq. (17)，我们直接 Diffusion 的 $\log\pi_\theta$、$\log\pi_{ref}$ 带入会得到：

$$\begin{aligned}&-\mathcal{L}_{KL_{sampling}}\\&=\frac{\beta}{2}\mathbb{E}_{\boldsymbol{c}\sim D,t\sim U(0,T)}\left(\frac{\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{\theta}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2}{2\sigma_t^2}-\frac{\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{ref}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2}{2\sigma_t^2}\right)^2\end{aligned}\tag{64}$$

图 3: kl 是直接期望的 KL，k2$_k l$　　KL

以上两种基于采样的 KL loss 是完全等价的，它们的梯度表达式均为：

$$
\begin{aligned}
&-\nabla_{\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,\boldsymbol{c},t)}\mathcal{L}_{KL_{sampling}} \\
&= \frac{\beta}{4\sigma_t^4}\mathbb{E}_{\boldsymbol{c}\sim D,t\sim U(0,T)}\Big[\left(\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{ref}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2-\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{\theta_{\text{old}}}\left(\boldsymbol{x}_t,\boldsymbol{c},t\right)\|^2\right) \\
&\qquad\qquad \cdot\nabla_{\boldsymbol{\mu}_\theta}\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\|^2\Big] \\
&= \frac{\beta}{4\sigma_t^4}\mathbb{E}\Big[SG\left(\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{ref}\|^2-\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\|^2\right)\left(\boldsymbol{\mu}_\theta-\boldsymbol{x}_{t-1}\right)\Big]
\end{aligned}
\tag{65}
$$

如果样本量非常小，比如单样本就更新模型参数，可以构造反例：当 $\boldsymbol{\mu}_{ref}>\boldsymbol{x}_{t-1}>\boldsymbol{\mu}_\theta$ 且 $\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_{ref}\|^2>\|\boldsymbol{x}_{t-1}-\boldsymbol{\mu}_\theta\|^2$ 时，可得 $-\nabla_{\boldsymbol{\mu}_\theta}\mathcal{L}_{DDPO_{wrong_{KL}}}<0$。这表明即使 $\boldsymbol{\mu}_{ref}>\boldsymbol{\mu}_\theta$，$\boldsymbol{\mu}_\theta$ 仍会朝减小的方向更新，存在明显逻辑悖论。而在 LLM 中反例是不存在的，采样估计的问题在 Diffusion 中应该比 LLM 还严峻。

如果样本量特别大，参考 appendix B, 期望上会等价于 eq. (60)。我们必须强调，在 Diffusion 中，由于图片是非常高维度的，显存占用巨大，把样本量调大是很困难的，比如 [1] 的 batch_size 就非常小，完全不能和 LLM 相比。所以非常推荐在做 Diffusion RLHF 中使用我们提出的 eq. (61)。相比 [14] 文章中加入 Diffusion Pretrain 的 loss，它也有着优势，因为加入 Pretrain loss 必须准备图文数据集，而我们的方法只需要准备文本绘图指令即可。同时 Pretrain loss 可能会因为预训练数据集和强化中准备的数据集

（比如自然图片和卡通图片）分布不同，引入分布的偏移，这是很不利的。

4.4 DiffusionDPO[12] 分析

直接向 LLM 的 DPO losseq. (105) 中带入 Diffuison 的 $\log \pi_\theta$、$\log \pi_{ref}$ 会得到：

$$
\begin{aligned}
L_{Diffusion_{DPO}}(\theta) = -\mathbb{E} \log \sigma \Bigg(-\beta \Bigg(\Big[\frac{\left\| x_{t-1}^w - \mu_\theta(\boldsymbol{x}_t^w, c, t) \right\|^2}{2\sigma_t^2} - \frac{\left\| x_{t-1}^w - \mu_{\text{ref}}(\boldsymbol{x}_t^w, c, t) \right\|^2}{2\sigma_t^2} \Big] \\
- \Big[\frac{\left\| x_{t-1}^l - \mu_\theta(\boldsymbol{x}_t^l, t) \right\|^2}{2\sigma_t^2} - \frac{\left\| x_{t-1}^l - \mu_{\text{ref}}(\boldsymbol{x}_t^l, t) \right\|^2}{2\sigma_t^2} \Big] \Bigg) \Bigg)
\end{aligned} \tag{66}
$$

在 DiffusionDPO 中具体基于 ϵ 来表示，也是等价的。

$$
\begin{aligned}
L(\theta) = &-\mathbb{E}_{(\boldsymbol{x}_0^w, \boldsymbol{x}_0^l) \sim \mathcal{D}, t \sim \mathcal{U}(0,T), \boldsymbol{x}_t^w \sim q(\boldsymbol{x}_t^w | \boldsymbol{x}_0^w), \boldsymbol{x}_t^l \sim q(\boldsymbol{x}_t^l | \boldsymbol{x}_0^l)} \\
&\log \sigma \Bigg(-\beta T \omega(\lambda_t) \Bigg(\left\| \boldsymbol{\epsilon}^w - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^w, t) \right\|_2^2 - \left\| \boldsymbol{\epsilon}^w - \boldsymbol{\epsilon}_{\text{ref}}(\boldsymbol{x}_t^w, t) \right\|_2^2 \\
&- \Big[\left\| \boldsymbol{\epsilon}^l - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t^l, t) \right\|_2^2 - \left\| \boldsymbol{\epsilon}^l - \boldsymbol{\epsilon}_{\text{ref}}(\boldsymbol{x}_t^l, t) \right\|_2^2 \Big] \Bigg) \Bigg)
\end{aligned} \tag{67}
$$

求 $L_{Diffusion_{DPO}}(\theta)$ 梯度可得：

$$
-\nabla_\theta L_{Diffusion_{DPO}}(\theta) = \frac{\beta}{\sigma_t^2} \mathbb{E}_{(x, x_{t-1}^w, x_{t-1}^l) \sim \mathcal{D}} \Bigg[\underbrace{\sigma(\hat{r}_\theta(c, x_{t-1}^l) - \hat{r}_\theta(c, x_{t-1}^w))}_{\text{higher weight when reward estimate is wrong}} \Bigg[\underbrace{(x_{t-1}^w - \mu_\theta(\boldsymbol{x}_t^w, c, t)) \nabla_\theta \mu_{\theta}}_{\text{increase likelihood of } x} \tag{68}
$$

并且

$$
\hat{r}_\theta(c, x_{t-1}) = \beta \left[-\frac{\left\| x_{t-1} - \mu_\theta(\boldsymbol{x}_t, c, t) \right\|^2}{2\sigma_t^2} + \frac{\left\| x_{t-1} - \mu_{\text{ref}}(\boldsymbol{x}_t, c, t) \right\|^2}{2\sigma_t^2} \right] \tag{69}
$$

我们容易得知 eq. (69) 也存在在单样本时 eq. (65) 中相同的反例，DiffusionDPO 的对有效采样样本量的需求同样是巨量的，而这在工程上是极难实现的，参考模型很难发挥应有的作用，有些工作干脆直接删掉了 [15]。

我认为与其这样不如直接将 eq. (61) 中的 $r(x_0, c)$ 正样本设置为正固定系数 A，负样本设置为-A。A 可以通过搜参来实现。

参考文献

[1] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

[2] dhcode cpp. X-r1, 2025.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[4] Jian Hu, Xibin Wu, Zilin Zhu, Weixun Wang, Dehao Zhang, Yu Cao, et al. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

[5] Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025.

[6] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[7] John Schulman. Approximating kl divergence, 2020.

[8] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[9] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.

[10] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022.

[11] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

[12] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.

[13] Hongyu Zang. The critical implementation detail of kl loss in grpo, 2025. Notion Blog.

[14] Yinan Zhang, Eric Tzeng, Yilun Du, and Dmitry Kislyuk. Large-scale reinforcement learning for diffusion models. In *European Conference on Computer Vision*, pages 1–17. Springer, 2024.

[15] Zhenglin Zhou, Xiaobo Xia, Fan Ma, Hehe Fan, Yi Yang, and Tat-Seng Chua. Dreamdpo: Aligning text-to-3d generation with human preferences via direct preference optimization. *arXiv preprint arXiv:2502.04370*, 2025.

[16] 蜡笔小熊猫. 为什么 dpo 的 chosen reward 会和 reject reward 一起降低？, 2024.

A 分类模型 KL 散度梯度推导

A.1 KL 散度的定义

KL 散度衡量两个离散概率分布 π_θ 和 π_{ref} 之间的差异，定义为：

$$\begin{aligned}\mathrm{KL}(\pi_\theta(y|x)\|\pi_{ref}(y|x)) &= \mathbb{E}_{x\sim D, y\sim\pi_\theta(y|x)}\left[\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}\right] \\ &= \mathbb{E}_{x\sim D}\sum_{y\in\mathcal{Y}}\left(\pi_\theta(y|x)\cdot\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}\right),\end{aligned} \tag{70}$$

其中 $\mathcal{Y}$ 为离散类别空间。

A.2 KL 散度梯度推导步骤

步骤 1：展开 KL 表达式 直接写出离散求和形式：

$$\mathrm{KL}\big(\pi_\theta(\cdot|x)\|\pi_{ref}(\cdot|x)\big) = \sum_y \pi_\theta(y|x)\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}. \tag{71}$$

步骤 2：应用梯度算子 对 θ 求梯度：

$$-\nabla_\theta\mathrm{KL}\big(\pi_\theta(\cdot|x)\|\pi_{ref}(\cdot|x)\big) = -\nabla_\theta\sum_y \pi_\theta(y|x)\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}. \tag{72}$$

步骤 3：交换求和与梯度 由于求和项有限且 $\pi_\theta(y|x)$ 光滑，可交换求和与梯度：

$$-\nabla_\theta\mathrm{KL}\big(\pi_\theta(\cdot|x)\|\pi_{ref}(\cdot|x)\big) = -\sum_y \nabla_\theta\left[\pi_\theta(y|x)\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}\right]. \tag{73}$$

步骤 4：乘积法则分解 对每一项应用乘积法则：

$$\begin{aligned}&-\nabla_\theta\left[\pi_\theta(y|x)\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}\right] \\ &\quad= -[\underbrace{\nabla_\theta\pi_\theta(y|x)\cdot\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}}_{\text{Term 1}} + \underbrace{\pi_\theta(y|x)\cdot\nabla_\theta\log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}}_{\text{Term 2}}].\end{aligned} \tag{74}$$

步骤 5：简化 Term 2 由于 $\pi_{ref}(y|x)$ 与 θ 无关，有：

$$\nabla_\theta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} = \nabla_\theta \log \pi_\theta(y|x) = \frac{\nabla_\theta \pi_\theta(y|x)}{\pi_\theta(y|x)}. \tag{75}$$

因此 Term 2 简化为：

$$\pi_\theta(y|x) \cdot \frac{\nabla_\theta \pi_\theta(y|x)}{\pi_\theta(y|x)} = \nabla_\theta \pi_\theta(y|x). \tag{76}$$

步骤 6：合并两项 将 Term 1 和 Term 2 相加：

$$\sum_y \left[\nabla_\theta \pi_\theta(y|x) \cdot \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} + \nabla_\theta \pi_\theta(y|x) \right]. \tag{77}$$

步骤 7：处理归一化条件 由于 $\sum_y \pi_\theta(y|x) = 1$，其梯度为 0：

$$\sum_y \nabla_\theta \pi_\theta(y|x) = \nabla_\theta \sum_y \pi_\theta(y|x) = \nabla_\theta 1 = 0. \tag{78}$$

因此第二项求和为 0，仅保留第一项：

$$-\nabla_\theta \mathrm{KL}\big(\pi_\theta(\cdot|x) \| \pi_{ref}(\cdot|x)\big) = -\sum_y \nabla_\theta \pi_\theta(y|x) \cdot \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}. \tag{79}$$

步骤 8：对数导数技巧 利用 $\nabla_\theta \pi_\theta(y|x) = \pi_\theta(y|x) \nabla_\theta \log \pi_\theta(y|x)$，改写为：

$$-\nabla_\theta \mathrm{KL}\big(\pi_\theta(\cdot|x) \| \pi_{ref}(\cdot|x)\big) = -\sum_y \pi_\theta(y|x) \nabla_\theta \log \pi_\theta(y|x) \cdot \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}. \tag{80}$$

步骤 9：期望形式 最终梯度可表示为期望：

$$-\nabla_\theta \mathrm{KL}(\pi_\theta(\cdot|x) \| \pi_{ref}(\cdot|x)) = -\mathbb{E}_{x\sim D, y\sim\pi_\theta(y|x)} \left[\nabla_\theta \log \pi_\theta(y|x) \cdot \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} \right]. \tag{81}$$

B 高维高斯分布的 KL 散度和梯度的推导

考虑两个 k 维各向同性高斯分布：$P \sim \mathcal{N}(\boldsymbol{\mu}_\theta, \sigma_1^2 \mathbf{I})$, $Q \sim \mathcal{N}(\boldsymbol{\mu}_{\text{ref}}, \sigma_2^2 \mathbf{I})$
其概率密度函数分别为：

$$\begin{aligned} P(\mathbf{x}) &= (2\pi\sigma_1^2)^{-k/2} \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_\theta\|^2}{2\sigma_1^2} \right) \\ Q(\mathbf{x}) &= (2\pi\sigma_2^2)^{-k/2} \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_{\text{ref}}\|^2}{2\sigma_2^2} \right) \end{aligned} \tag{82}$$

B.1 KL 散度定义

KL 散度定义为：

$$D_{\mathrm{KL}}(P \parallel Q) = \mathbb{E}_{\mathbf{x}\sim P}\left[\ln \frac{P(\mathbf{x})}{Q(\mathbf{x})}\right] \tag{83}$$

将密度函数代入，展开对数比：

$$\ln \frac{P(\mathbf{x})}{Q(\mathbf{x})} = \frac{k}{2}\ln\frac{\sigma_2^2}{\sigma_1^2} + \left[-\frac{\|\mathbf{x}-\boldsymbol{\mu}_\theta\|^2}{2\sigma_1^2} + \frac{\|\mathbf{x}-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}\right] \tag{84}$$

B.2 逐项计算期望

将 KL 散度拆分为常数项与二次项之和：

$$D_{\mathrm{KL}} = \underbrace{\frac{k}{2}\ln\frac{\sigma_2^2}{\sigma_1^2}}_{\text{常数项}} + \underbrace{\mathbb{E}_P\left[-\frac{\|\mathbf{x}-\boldsymbol{\mu}_\theta\|^2}{2\sigma_1^2} + \frac{\|\mathbf{x}-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}\right]}_{\text{二次项}} \tag{85}$$

计算 $\mathbb{E}_P[\|\mathbf{x}-\boldsymbol{\mu}_\theta\|^2]$　由于 $\mathbf{x}\sim P$，协方差矩阵为 $\sigma_1^2\mathbf{I}$，故：

$$\mathbb{E}_P[\|\mathbf{x}-\boldsymbol{\mu}_\theta\|^2] = \mathrm{tr}(\sigma_1^2\mathbf{I}) = k\sigma_1^2 \tag{86}$$

计算 $\mathbb{E}_P[\|\mathbf{x}-\boldsymbol{\mu}_{\mathbf{ref}}\|^2]$　展开二次型并取期望：

$$\begin{aligned}\|\mathbf{x}-\boldsymbol{\mu}_{\mathrm{ref}}\|^2 &= \|\mathbf{x}-\boldsymbol{\mu}_\theta+\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2 \\ &= \|\mathbf{x}-\boldsymbol{\mu}_\theta\|^2 + 2(\mathbf{x}-\boldsymbol{\mu}_\theta)^\top(\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}) + \|\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2\end{aligned} \tag{87}$$

取期望后，交叉项因 $\mathbb{E}_P[\mathbf{x}-\boldsymbol{\mu}_\theta]=0$ 而消失：

$$\mathbb{E}_P[\|\mathbf{x}-\boldsymbol{\mu}_{\mathrm{ref}}\|^2] = k\sigma_1^2 + \|\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2 \tag{88}$$

代入二次项

$$\begin{aligned}\mathbb{E}_P\left[-\frac{\|\mathbf{x}-\boldsymbol{\mu}_\theta\|^2}{2\sigma_1^2} + \frac{\|\mathbf{x}-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}\right] &= -\frac{k\sigma_1^2}{2\sigma_1^2} + \frac{k\sigma_1^2+\|\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2} \\ &= -\frac{k}{2} + \frac{k\sigma_1^2}{2\sigma_2^2} + \frac{\|\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}\end{aligned} \tag{89}$$

合并所有项　将常数项与二次项合并：

$$\begin{aligned}D_{\mathrm{KL}} &= \frac{k}{2}\ln\frac{\sigma_2^2}{\sigma_1^2} - \frac{k}{2} + \frac{k\sigma_1^2}{2\sigma_2^2} + \frac{\|\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2} \\ &= \frac{k}{2}\left[2\ln\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2}{\sigma_2^2} - 1\right] + \frac{\|\boldsymbol{\mu}_\theta-\boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}\end{aligned} \tag{90}$$

最终公式 整理后得到：

$$D_{\mathrm{KL}}(P \parallel Q) = \underbrace{k\left(\ln\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2}{2\sigma_2^2} - \frac{1}{2}\right)}_{\text{方差项}} + \underbrace{\frac{\|\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}}_{\text{均值项}} \tag{91}$$

当 $\sigma = \sigma_1 = \sigma_2$ 时，

$$D_{\mathrm{KL}}(P \parallel Q) = \frac{\|\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma^2} \tag{92}$$

B.3 对 $\boldsymbol{\mu}_\theta$ 的梯度推导

从 KL 散度公式中提取与 $\boldsymbol{\mu}_\theta$ 相关的项：

$$D_{\mathrm{KL}}(P \parallel Q) = \underbrace{\frac{\|\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}\|^2}{2\sigma_2^2}}_{\text{唯一与 } \boldsymbol{\mu}_\theta \text{ 相关的项}} + \text{其他与 } \boldsymbol{\mu}_\theta \text{ 无关的项.} \tag{93}$$

步骤 1：展开二次型 对 $\|\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}\|^2$ 展开：

$$\|\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}\|^2 = (\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}})^\top(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}). \tag{94}$$

步骤 2：计算关于 $\boldsymbol{\mu}_\theta$ 的梯度 利用二次型的梯度公式：

$$-\nabla_{\boldsymbol{\mu}_\theta}\left[(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}})^\top(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}})\right] = -2(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}). \tag{95}$$

步骤 3：组合梯度分量 将梯度结果代入 KL 散度表达式：

$$-\nabla_{\boldsymbol{\mu}_\theta} D_{\mathrm{KL}} = -\frac{1}{2\sigma_2^2} \cdot 2(\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}) = -\frac{\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}}{\sigma_2^2}. \tag{96}$$

最终梯度表达式

$$-\nabla_{\boldsymbol{\mu}_\theta} D_{\mathrm{KL}}(P \parallel Q) = -\frac{\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\mathrm{ref}}}{\sigma_2^2} \tag{97}$$

C DPO 推导

基于 KL 约束的奖励最大化目标，推导出可操作的直接偏好优化目标函数 [6]。首先建立基础优化问题：

$$
\begin{aligned}
&\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi}[r(x,y)] - \beta D_{\mathrm{KL}}[\pi(y|x)\|\pi_{\mathrm{ref}}(y|x)] \\
&= \max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[r(x,y) - \beta\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right] \\
&= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \frac{1}{\beta}r(x,y)\right] \\
&= \min_{\pi}\mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]
\end{aligned}
\tag{98}
$$

配分函数 $Z(x)$ 构造概率分布：

$$
Z(x) = \sum_{y}\pi_{\mathrm{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{99}
$$

定义新的参考分布 π^* 为：

$$
\pi^*(y|x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{100}
$$

将目标函数重构为：

$$
\begin{aligned}
&\min_{\pi}\mathbb{E}_{x\sim\mathcal{D},y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi^*(y|x)} - \log Z(x)\right] \\
&= \min_{\pi}\mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi^*(y|x)}\right] - \log Z(x)\right] \\
&= \min_{\pi}\mathbb{E}_{x\sim\mathcal{D}}\left[D_{KL}\left(\pi(y|x)\parallel\pi^*(y|x)\right) - \log Z(x)\right]
\end{aligned}
\tag{101}
$$

由于 $Z(x)$ 与策略 π 无关，优化目标简化为最小化 KL 散度项。根据 KL 散度的非负性，当 $\pi = \pi^*$ 时取得全局最优解。

C.1 从奖励建模到偏好学习

实际应用中直接求解 π^* 存在两大障碍：1) 真实奖励函数 r^* 未知；2) 配分函数 $Z(x)$ 的计算涉及全响应空间积分。为此，我们引入偏好学习框架。

采用 Bradley-Terry 模型，对于输入 x 和响应对 (y_w, y_l)，偏好概率建模为：

$$
p^*(y_w \succ y_l|x) = \frac{\exp(r^*(x,y_w))}{\exp(r^*(x,y_w)) + \exp(r^*(x,y_l))} \tag{102}
$$

关键突破在于建立奖励函数与最优策略的显式关联。由式 (2) 可得：

$$r(x,y) = \beta \log \frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x) \tag{103}$$

将奖励差表达式代入偏好概率模型：

$$p^*(y_w \succ y_l|x) = \frac{1}{1+\exp\left(\beta \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)} - \beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)}\right)} \tag{104}$$

C.2 最终目标函数

通过极大似然估计，得到直接优化策略的参数化目标函数：

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right] \tag{105}$$

C.3 DPO 梯度

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$
$$-\beta \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\underbrace{\sigma(\hat{r}_\theta(x,y_l) - \hat{r}_\theta(x,y_w))}_{\text{higher weight when reward estimate is wrong}}\left[\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l \mid x)}_{\text{decrease likelihood of } y_l}\right]\right]. \tag{106}$$

D 重要性采样与 PPO-2 clip

D.1 基本概念与动机

重要性采样（Importance Sampling）是强化学习中实现离策略（off-policy）学习的关键技术，其核心思想是通过引入行为策略（behavior policy）的采样分布来估计目标策略（target policy）的期望值。这一方法在策略优化中具有双重意义：

1. 样本复用：允许利用历史策略生成的旧样本进行当前策略更新，显著提升数据利用率 2. 方差控制：通过重要性权重修正新旧策略的概率分布偏差，维持无偏估计特性

D.2 策略梯度推导

考虑策略梯度基本形式：

$$\nabla_\theta J(\theta) = \mathbb{E}_{,y\sim\pi_\theta}\left[A(x,y)\nabla_\theta \log \pi_\theta(y|x)\right] \tag{107}$$

当转换为离策略更新时，需要引入重要性权重（Importance Weight）$\rho_t = \frac{\pi_\theta(y|x)}{\pi_{\theta'}(y|x)}$：

$$\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{(x,y)\sim\pi_{\theta'}}\left[\frac{\pi_\theta(x,y)}{\pi_{\theta'}(x,y)} A_\theta(x,y)\nabla_\theta \log \pi_\theta(y|x)\right] \\
&= \mathbb{E}_{(x,y)\sim\pi_{\theta'}}\left[\frac{\pi_\theta(y|x)\cancel{\pi_\theta(x)}}{\pi_{\theta'}(y|x)\cancel{\pi_{\theta'}(x)}} A_\theta(x,y)\nabla_\theta \log \pi_\theta(y|x)\right] \\
&\approx \mathbb{E}_{(x,y)\sim\pi_{\theta'}}\left[\frac{\pi_\theta(y|x)}{\pi_{\theta'}(y|x)} A_{\theta'}(x,y)\nabla_\theta \log \pi_\theta(y|x)\right]
\end{aligned} \tag{108}$$

推导过程中包含两个重要近似：

- 状态分布抵消假设：$\pi_\theta(s_t) \approx \pi_{\theta'}(s_t)$，在策略更新幅度较小时成立
- 优势函数近似：$A_\theta(x,y) \approx A_{\theta'}(x,y)$，要求新旧策略差异可控

D.3 目标函数形式化

令 $\theta' = \theta_{\text{old}}$，得到离策略目标函数：

$$J(\theta) = \mathbb{E}_{(x,y)\sim\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta(y|x)}{\pi_{\theta_{\text{old}}}(y|x)} A_{\theta_{\text{old}}}(x,y)\right] \tag{109}$$

该目标函数具有以下特性：

1. **无偏性**：当 π_θ 与 $\pi_{\theta_{\text{old}}}$ 的支撑集相同时保持无偏估计
2. **方差敏感性**：重要性权重 ρ_t 的数值稳定性直接影响梯度质量
3. **策略约束**：需通过 KL 散度等度量限制 π_θ 与 $\pi_{\theta_{\text{old}}}$ 的差异

D.4 PPO-2 clip 方法

针对重要性采样方差问题，PPO-2 clip 提出带截断的替代目标函数：

$$J^{\text{clip}}(\theta) = \mathbb{E}_t \left[\min \left(\rho_t A_{\theta_{\text{old}}}(x, y),\ \text{clip}(\rho_t, 1-\epsilon, 1+\epsilon) A_{\theta_{\text{old}}}(x, y) \right) \right] \quad (110)$$

其中 $\rho_t = \frac{\pi_\theta(y|x)}{\pi_{\theta_{\text{old}}}(y|x)}$，$\epsilon$ 为截断阈值（通常取 0.1-0.2）。该方法通过双向截断实现：

1. 当 $A_t > 0$ 时，限制策略提升幅度不超过 $1+\epsilon$ 2. 当 $A_t < 0$ 时，限制策略下降幅度不超过 $1-\epsilon$

核心优势：

- **梯度稳定**：通过硬性截断避免过大策略更新
- **计算高效**：无需计算 KL 散度等附加约束项
- **兼容优势**：兼容 GAE 等多种优势估计方法

该设计在保证样本效率的同时，显著提升了策略优化的稳定性，成为深度强化学习领域最广泛应用的算法之一。

E k3 估计不存在绝对优势

Schulman, John [7] 认为 k3 估计相比 k1、k2 估计存在绝对优势，有着更小的偏差和方差。但即便是在纯 KL 散度估计领域，其实也并不成立。相比原始实现 [7]，下面代码对 truekl 的计算进行了修正：

```python
import torch.distributions as dis
import torch
p = dis.Normal(loc=0, scale=1)
q = dis.Normal(loc=0.1, scale=0.2)
x = q.sample(sample_shape=(10_000,))
truekl = dis.kl_divergence(q, p)
print("true", truekl)
logr = p.log_prob(x) - q.log_prob(x)
k1 = -logr
k2 = logr ** 2 / 2
k3 = (logr.exp() - 1) - logr
k4 = torch.where(k1 < 0, k3, torch.minimum(k1, k3))
r = torch.exp(logr)
clip_r = torch.clamp(r,max=10)
k3_clip = clip_r - 1 - logr
for k in (k1, k2, k3, k4, k3_clip):
    print((k.mean() - truekl) / truekl, k.std() / truekl)
```

运行结果为：

```
truekl tensor(1.1344)
tensor(0.0007) tensor(0.5896)
tensor(-0.2348) tensor(0.4524)
tensor(-0.4051) tensor(2.4417)
tensor(-0.4051) tensor(2.4417)
tensor(-0.4676) tensor(0.4370)
```

k3 估计的方差是远大于 k1 估计和真实值的，这主要源于它进行了 exp() 计算，为此我们提出 k3_clip，限制 exp() 的最大值，避免数值非线性增长带来的问题。

F GROUP Norm 方差一定 <1

“一个列表，每个元素都是从 0 - 1 间取的，列表中元素的方差可能大于 1 吗”？

列表中元素在 0 到 1 之间取值时，方差不可能大于 1。接着阐述了方差的定义，设列表为 $x_1, x_2, \cdots, x_n$ ，均值 $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$，方差计算公式为 $Var(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$。

推理过程如下：- 因为 $0 \leq x_i \leq 1$，所以均值 $0 \leq \bar{x} \leq 1$。- 由此推出 $(x_i - \bar{x})$ 的取值范围是 $-1 \leq x_i - \bar{x} \leq 1$ ，平方后得到 $0 \leq (x_i - \bar{x})^2 \leq 1$ 。- 又因为方差 $Var(X)$ 是 $(x_i - \bar{x})^2$ 的平均值，所以 $0 \leq Var(X) \leq 1$。

G KL 梯度系数可视化代码

```python
import torch
import matplotlib.pyplot as plt

# 设定一个范围用于绘制曲线
pertokenlogps = torch.linspace(-3, 2, steps=400)

# 计算每条曲线的值
curve2 = pertokenlogps + 0.7
curve3 = 1 - torch.exp(-0.7 - pertokenlogps)
curve4 = torch.exp(pertokenlogps) - torch.exp(torch.tensor(-0.7))
# 绘制曲线
plt.figure(figsize=(10, 6))
plt.plot(pertokenlogps, curve2, label='pertokenlogps - (-0.7)  (k1/k2)')
plt.plot(pertokenlogps, curve3, label='1 - exp(-0.7 - pertokenlogps)   (k3)')
```

```
plt.plot(pertokenlogps, curve4, label='exp(pertokenlogps) - exp(-0.7) minimax')
plt.xlabel('pertokenlogps')
plt.ylabel('Value')
plt.title('set log \pi_ref = -0.7, as log(0.5) = -0.69.')
plt.legend()
plt.grid(True)
plt.show()
```